CLAIMS

WHAT IS CLAIMED:

1. A pattern generator, comprising:

an address generator adapted to provide a first address having a plurality of address bits;

an address topology generator including a first plurality of programmable logic gates, each programmable logic gate of the first plurality coupled to receive at least a subset of the plurality of address bits, the first plurality of programmable logic gates generating a second address having a plurality of modified address bits; and

a data topology generator adapted to receive at least a subset of the plurality of modified address bits and generate write data based on the subset of modified address bits.

2. The pattern generator of claim 1, wherein the address topology generator further includes:

an inversion register for storing a plurality of inversion bits, wherein at least a subset of the first plurality of programmable logic gates have an associated inversion bit coupled thereto for inverting the outputs of the first subset of programmable logic gates.

3. The pattern generator of claim 1, wherein the address topology generator further includes:

a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of

address bits, the second plurality of programmable logic gates generating a lookup address; and

a lookup table having a lookup table output and being adapted to receive the lookup address and provide lookup data based on the lookup address.

4. The pattern generator of claim 3, wherein the address topology generator further includes a first logic gate having a first input coupled to the lookup table, a second input coupled to at least one of the first subset of programmable gates, and an output, at least one of the modified address bits being based on the output of the first logic gate.

5. The pattern generator of claim 4, wherein the first logic gate comprises an exclusive OR logic gate.

6. The pattern generator of claim 4, wherein the address topology generator further includes:

a lookup table enable register for storing a plurality of lookup enable bits; and

a second logic gate having a first input coupled to a lookup table bit of the lookup table enable register, a second input being coupled to the lookup table, and an output coupled to the first logic gate.

7. The pattern generator of claim 4, wherein the second logic gate comprises an AND logic gate.

8. The pattern generator of claim 1, wherein the address topology generator further includes a select register for storing a plurality of select bits, the select register being coupled to the first plurality of programmable gates for masking ones of the subset of the plurality of address bits.

9. The pattern generator of claim 3, wherein the address topology generator further includes a select register for storing a plurality of select bits, the select register being coupled to the second plurality of programmable gates for masking ones of the subset of the plurality of address bits.

10. The pattern generator of claim 1, wherein the data topology generator is further adapted to generate expected data based on the subset of the plurality of modified address bits.

11. The pattern generator of claim 1, wherein the data topology generator includes:

a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of modified address bits, the second plurality of programmable logic gates generating a lookup address; and

a lookup table having a lookup table output and being adapted to receive the lookup address and provide lookup data based on the lookup address.

12. The pattern generator of claim 11, wherein the data topology generator further includes:

a control register for storing control bits; and

a first logic gate having a first input coupled to the lookup table, a second input coupled to a control bit of the control register, and an output, the write data being based on the output of the first logic gate.

13. The pattern generator of claim 11, wherein the data topology generator further includes a bypass programmable gate having an output and being coupled to receive at least a subset of the plurality of modified address bits, the write data being based the output of the bypass programmable gate.

14. The pattern generator of claim 11, wherein the data topology generator further includes a select register for storing a plurality of select bits, the select register being coupled to the second plurality of programmable gates for masking ones of the subset of the plurality of modified address bits.

15. The pattern generator of claim 11, wherein the data topology generator further includes an inversion register for storing a plurality of inversion bits, wherein at least a subset of the second plurality of programmable gates have an associated inversion bit coupled thereto for inverting the outputs of the second subset of programmable gates.

16. The pattern generator of claim 1, further comprising:

a micro-sequencer adapted to receive test instructions and change the value of the first address in the address generator based on the test instructions.

17. The pattern generator of claim 1, wherein the data topology generator is adapted to receive an inversion input and invert the write data based on the inversion input.

18. The pattern generator of claim 10, wherein the data topology generator is adapted to receive an inversion input and invert the expected data based on the inversion input.

19. The pattern generator of claim 1, wherein the programmable logic gates comprise exclusive OR logic gates.

20. The pattern generator of claim 1, further comprising:

a microprocessor adapted to control the address generator to update the first address based on a plurality of instructions, the microsequencer including:

an instruction queue indexed by an instruction pointer and adapted to store instructions for updating the first address, the instructions capable of being nested in a plurality of loop levels;

a plurality of loop counters adapted to store a remaining number of iterations associated with an associated loop level; and

a plurality of loop instruction pointer registers associated with the loop counter and adapted to store a loop instruction pointer value

associated with a particular one of the instructions marking the start of the associated loop level, wherein the microsequencer is adapted to load the instruction pointer with the loop instruction pointer value associated with the loop level in response to the remaining number of iterations for the loop level equaling zero.

21. The pattern generator of claim 20, wherein the microsequencer further includes a plurality of loop count registers adapted to store a total number of iterations associated with an associated loop level, and the microsequencer is adapted to load the loop counter associated with the loop level with the total number of iterations when the particular one of the instructions marking the start of the associated loop level is encountered.

22. The pattern generator of claim 20, wherein the microsequencer further includes a repeat counter adapted to store a repeat value indicating a number of remaining repeat iterations of a selected one of the instructions.

23. The pattern generator of claim 22, wherein the microsequencer further comprises a repeat count register adapted to store a total number of repeat iterations associated with the selected one of the instructions.

24. The pattern generator of claim 20, wherein the address generator includes a plurality of address registers and the microsequencer is adapted to update the address register based on the instructions.

25. A system for testing a device having an address topology and a data topology, comprising:

a microprocessor adapted to provide a plurality of test instructions;

a pattern generator adapted to receive the test instructions and provide a second address for accessing the device, the second address having a plurality of modified address bits and being based on the address topology of the device, the pattern generator comprising:

an address generator adapted to provide a first address, the first address having a plurality of address bits;

a micro-sequencer adapted to receive the test instructions and update the first address based on the test instructions;

an address topology generator including a first plurality of programmable logic gates, each programmable logic gate of the first plurality coupled to receive at least a subset of the plurality of address bits, the first plurality of programmable logic gates generating the second address having a plurality of modified address bits, the second address being a logical combination of the subset of the plurality of address bits, the logical combination being based on the address topology of the device; and

a data topology generator adapted to receive at least a subset of the plurality of modified address bits and generate write data based on

the subset of the modified address bits and the data topology of the device; and

a timing signal generator adapted to generate timing signals for accessing the device.

26. The system of claim 25, further comprising:

a multiplexer coupled to the pattern generator and adapted to receive first and second subsets of the second address, the first subset defining a row address and the second subset defining a column address, wherein an output of the multiplexer is coupled to the device for providing one of the row address and the column address to access the device.

27. The system of claim 26, wherein the multiplexer has a control input coupled to the timing signal generator, the timing signal generator being adapted to select one of the row address and the column address based on the timing signals.

28. The system of claim 25, wherein the data topology generator is further adapted to generate expected data based on the subset of the plurality of modified address bits.

29. The system of claim 28, further comprising:

a comparator adapted to receive the expected data and output data from the device, compare the expected data and output data, and provide a test indicator to the microprocessor based on the comparison.

30. The system of claim 25, wherein the address topology generator further includes:

a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of address bits, the second plurality of programmable logic gates generating a lookup address; and

a lookup table having a lookup table output and being adapted to receive the lookup address and provide lookup data based on the lookup address.

31. The system of claim 25, wherein the data topology generator includes:

a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of modified address bits, the second plurality of programmable logic gates generating a lookup address; and

a lookup table having a lookup table output and being adapted to receive the lookup address and provide lookup data based on the lookup address.

32. The system of claim 25, wherein the data topology generator further includes:

a control register for storing control bits; and

a first logic gate having a first input coupled to the lookup table, a second input coupled to a control bit of the control register, and an output, the write data being based on the output of the first logic gate.

33. The system of claim 25, wherein the data topology generator further includes a bypass programmable gate having an output and being coupled to receive at least a subset of the plurality of modified address bits, the write data being based the output of the bypass programmable gate.

34. The system of claim 25, wherein the data topology generator is adapted to receive an inversion input and invert the write data based on the inversion input.

35. The pattern generator of claim 28, wherein the data topology generator is adapted to receive an inversion input and invert the expected data based on the inversion input.

36. The pattern generator of claim 25, wherein the programmable logic gates comprise exclusive OR logic gates.

37. A method for generating a pattern, comprising:

generating a first address having a plurality of address bits;

generating a second address having a plurality of modified address bits, the second address comprising a programmable combination of subsets of the address bits; and

generating write data based on a subset of the modified address bits.

38. The method of claim 37, wherein generating the second address includes inverting at least one of the modified address bits.

39. The method of claim 37, wherein generating the second address includes:

generating a lookup address based on a programmable combination of subsets of the address bits; and

accessing a lookup table with the lookup address to generate lookup data; and

generating at least one of the modified address bits based on the lookup data.

40. The method of claim 37, further comprising generating expected data based on the subset of the plurality of modified address bits.

41. The method of claim 37, wherein generating the write data includes:

generating a lookup address based on a programmable combination of subsets of the modified address bits; and

accessing a lookup table with the lookup address to generate lookup data; and

generating at least a portion of the write data based on the lookup data.

42. The method of claim 37, further comprising inverting the write data.

43. The method of claim 37, further comprising providing a plurality of instructions, wherein generating the first address includes generating the first address based on the instructions.

44. The method of claim 43, wherein providing the instructions includes:

storing the plurality of instructions in an instruction queue indexed by an instruction pointer;

identifying a first instruction marking the start of a first loop, the first instruction having a first loop instruction pointer value;

storing a remaining number of iterations for the first loop in a first loop counter; and

loading the instruction pointer with the first loop instruction pointer value in response to the remaining number of iterations for the first loop equaling zero.

45. The method of claim 44, further comprising storing the first instruction pointer value in a first loop instruction pointer register.

46. The method of claim 44, further comprising:

identifying a second instruction nested within the first loop and marking the start of a second loop, the second instruction having a second loop instruction pointer value;

storing a remaining number of iterations for the second loop in a second loop counter; and

loading the instruction pointer with the second loop instruction pointer value in response to the remaining number of iterations for the second loop equaling zero.

47. The method of claim 44, further storing a total number of iterations associated with the first loop in a first loop count register.

48. The method of claim 47, further comprising:

identifying a second instruction marking the end of the first loop; and

loading the first loop counter with the total number of iterations when the second instruction is first encountered.

49. The method of claim 44, further comprising:

identifying a second instruction to be repeated;

storing a repeat value indicating a number of remaining repeat iterations for the second instruction in a repeat counter; and

repeating the second instruction and decrementing the repeat counter until the number of remaining repeat iterations for the second instruction equals zero.

50. The method of claim 44, further comprising:

storing a total number of repeat iterations associated with the second instruction in a repeat count register; and

loading the repeat counter with the total number of iterations in response to first encountering the second instruction.

51. A method for testing a device having an address topology and a data topology, comprising:

providing a plurality of test instructions;

providing a first address having a plurality of address bits;

updating the first address based on the test instructions;

programming a first plurality of programmable logic gates, each programmable logic gate of the first plurality coupled to receive at least a subset of the plurality of address bits, the first plurality of programmable logic gates generating a second address having a plurality of modified address bits, the second address being a logical combination of the first address, the logical combination being based on the address topology of the device; and

generating write data based on at least a subset of the plurality of modified address bits and the data topology of the device;

generating timing signals for accessing the device; and

accessing the device based on the second address and the write data.

52. The method of claim 51, further comprising:

defining a row address based on a first subset of the second address and a column address based on a second subset of the second address;

selecting one of the row address and the column address to access the device.

53. The method of claim 52, further comprising selecting one of the row address and the column address based on the timing signals.

54. The method of claim 51, further comprising generating expected data based on the subset of the plurality of modified address bits.

55. The method of claim 54, further comprising:

receiving output data from the device;

comparing the expected data and output data; and

providing a test indicator based on the comparison.

56. The method of claim 51, further comprising:

programming a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of address bits, the second plurality of programmable logic gates generating a lookup address;

accessing a lookup table based on the lookup address to provide lookup data; and

generating at least one of the modified address bits based on the lookup data.

57. The method of claim 51, further comprising:

programming a second plurality of programmable logic gates, each programmable logic gate of the second plurality being coupled to receive at least a subset of the plurality of modified address bits, the second plurality of programmable logic gates generating a lookup address;

accessing a lookup table based on the lookup address to provide lookup data; and

generating at least a portion of the write data based on the lookup data.

58. A pattern generator, comprising:

an address generator adapted to provide a first address having a plurality of address bits;

a microsequencer adapted to control the address generator to update the first address, the microsequencer including:

an instruction queue indexed by an instruction pointer and adapted to store instructions for updating the first address, the instructions capable of being nested in a plurality of loop levels;

a plurality of loop counters adapted to store a remaining number of iterations associated with an associated loop level; and

a plurality of loop instruction pointer registers associated with the loop counter and adapted to store a loop instruction pointer value associated with a particular one of the instructions marking the start of the associated loop level, wherein the microsequencer is adapted to load the instruction pointer with the loop instruction pointer value associated with the loop level in response to the remaining number of iterations for the loop level equaling zero;

an address topology generator adapted to receive at least a subset of the plurality of address bits and generate a second address having a plurality of modified address bits based on a logical combination of the address bits; and

a data topology generator adapted to receive at least a subset of the plurality of modified address bits and generate write data based on the subset of modified address bits.

59. The pattern generator of claim 58, wherein the microsequencer further includes a plurality of loop count registers adapted to store a total number of iterations associated with an associated loop level, and the microsequencer is adapted to load the loop counter associated with

the loop level with the total number of iterations when the particular one of the instructions marking the start of the associated loop level is encountered.

60. The pattern generator of claim 58, wherein the microsequencer further includes a repeat counter adapted to store a repeat value indicating a number of remaining repeat iterations of a selected one of the instructions.

61. The pattern generator of claim 60, wherein the microsequencer further comprises a repeat count register adapted to store a total number of repeat iterations associated with the selected one of the instructions.

62. The pattern generator of claim 58, wherein the address generator includes a plurality of address registers and the microsequencer is adapted to update the address register based on the instructions.

63. A microsequencer, comprising:

an instruction queue indexed by an instruction pointer and adapted to store instructions, the instructions capable of being nested in a plurality of loop levels;

a plurality of loop counters adapted to store a remaining number of iterations associated with an associated loop level; and

a plurality of loop instruction pointer registers associated with the loop counter and adapted to store a loop instruction pointer value associated with a particular one of the instructions marking the start of the associated loop level, wherein the

microsequencer is adapted to load the instruction pointer with the loop instruction pointer value associated with the loop level in response to the remaining number of iterations for the loop level equaling zero.

64. The microsequencer of claim 63, further comprising a plurality of loop count registers adapted to store a total number of iterations associated with an associated loop level, and the microsequencer is adapted to load the loop counter associated with the loop level with the total number of iterations when the particular one of the instructions marking the start of the associated loop level is encountered.

65. The microsequencer of claim 63, further comprising a repeat counter adapted to store a repeat value indicating a number of remaining repeat iterations of a selected one of the instructions.

66. The microsequencer of claim 65, further comprising a repeat count register adapted to store a total number of repeat iterations associated with the selected one of the instructions.

67. A method for executing instructions, comprising:

storing a plurality of instructions in an instruction queue indexed by an instruction pointer;

identifying a first instruction marking the start of a first loop, the first instruction having a first loop instruction pointer value;

storing a remaining number of iterations for the first loop in a first loop counter; and

loading the instruction pointer with the first loop instruction pointer value in response to the remaining number of iterations for the first loop equaling zero.

68. The method of claim 67, further comprising storing the first instruction pointer value in a first loop instruction pointer register.

69. The method of claim 67, further comprising:

identifying a second instruction nested within the first loop and marking the start of a second loop, the second instruction having a second loop instruction pointer value;

storing a remaining number of iterations for the second loop in a second loop counter; and

loading the instruction pointer with the second loop instruction pointer value in response to the remaining number of iterations for the second loop equaling zero.

70. The method of claim 67, further storing a total number of iterations associated with the first loop in a first loop count register.

71. The method of claim 70, further comprising:

identifying a second instruction marking the end of the first loop; and

loading the first loop counter with the total number of iterations when the second instruction is first encountered.

72. The method of claim 67, further comprising:

identifying a second instruction to be repeated;

storing a repeat value indicating a number of remaining repeat iterations for the second instruction in a repeat counter; and

repeating the second instruction and decrementing the repeat counter until the number of remaining repeat iterations for the second instruction equals zero.

73. The method of claim 67, further comprising:

storing a total number of repeat iterations associated with the second instruction in a repeat count register; and

loading the repeat counter with the total number of iterations in response to first encountering the second instruction.

74. A pattern generator, comprising:

means for generating a first address having a plurality of address bits;

means for generating a second address having a plurality of modified address bits, the second address comprising a programmable combination of subsets of the address bits; and

means for generating write data based on a subset of the modified address bits.

75. A microsequencer, comprising:

means for storing a plurality of instructions;

means for indexing the instructions using an instruction pointer;

means for identifying a first instruction marking the start of a first loop, the first instruction having a first loop instruction pointer value;

means for storing a remaining number of iterations for the first loop; and

means for loading the instruction pointer with the first loop instruction pointer value in response to the remaining number of iterations for the first loop equaling zero.